UUV Teams, Control From A Biological Perspective

P. McDowell
Naval Research Laboratory
Stennis Space Center, MS
39529-5004, USA
pmcdowell@nlrssc.navy.mil

J. Chen
Louisiana State University
Baton Rouge, LA
USA
jchen@lsu.edu

B. Bourgeois
Naval Research Laboratory
Stennis Space Center, MS
39529-5004, USA
bsb@nrlssc.navy.mil

Abstract-Remote Operated Vehicles (ROVs) are used extensively for salvage operations, ocean floor surveying and numerous inspection activities that support a wide range of underwater commercial activities. In deep water (greater than 1000 ft) an ROV is the platform of choice because of the depth and endurance limitations for human divers. The key disadvantage to an ROV is the requirement for the long tether. The tether greatly inhibits the speed of the ROV, requires a ship with deck gear capable of handling this cable, and significantly restricts ship movement while deployed.

Un-tethered Unmanned Underwater Vehicles (UUVs) have entered the commercial market and have demonstrated the ability to perform deep-water surveys faster and cheaper than towed vessels. With further technological advances, UUVs have the potential for supplementing and even replacing ROVs for many deep-water operations because of the cost and problems associated with the tether. One promising scenario for the near future is to use an ROV or surface ship to control multiple UUVs in a local work area. Typically in this scenario the UUVs are used to extend the sensor footprint of the ROV or surface ship.

Another area of interest is the UUV team concept. A stereotypical UUV team would be a heterogeneous mix of several low-cost specific purpose vehicles, guided and supported by one or two higher cost control vessels. Because of the severe restrictions that the sub-sea environment places on communication and positioning, precision underwater navigation is difficult. Currently most precision underwater navigation relies on some sort of infrastructure such as surface ships or underwater beacons placed in known positions. Using these assets as reference-points sub-sea navigation is carried out. Some situations require that the environmental and/or commercial attributes of an area be assessed before an infrastructure exists. In order to do this the UUV team must be able to navigate to an area, carry out its task and return without any pre-existing infrastructure or step by step guidance.

Given basic assumptions about the type and frequency of sensor input we present a biologically inspired, decentralized methodology for safely and efficiently moving a loose formation of UUV's to and from the task area with the goal of minimizing outside guidance.

I. Introduction

The PNT (Positioning, Navigation and Timing) team of the Naval Research Laboratory (NRL), located as Stennis Space Center, MS has recently undertaken a new project aimed at the conceptualization, development and testing of strategies, algorithms and sensors required to enable inter-vessel positioning and coordinated navigation needed for UUV (unmanned underwater vehicle) task forces. This effort supports an operational concept of UUVs that does not require the prior insertion of a communication and navigation infrastructure by ships or planes into the planned operational area.

The employment of multiple UUVs has significant advantages for both military and commercial applications [2]. Advantages include the ability to survey large ocean areas more rapidly and economically than can be accomplished with a ship or a single UUV, and increased operational envelopes due to weather insensitivity. Inter-vessel positioning and navigation has distinct advantages in that it can reduce or eliminate the requirement for pre-deployed positioning systems.

In this paper we describe methods that rely on machine learning techniques for creation and maintenance of loose formations of autonomous vehicles for the purpose of traveling from a mission staging area to an area of interest. Specifically, we have used a genetic algorithm (GA) to find relevant parameters for a feed-forward neural network that controls heading and speed changes for UUV's. Using these controllers and a leader/follower scheme we have achieved formation forming and maintaining behavior. We present a conceptual discussion and show simulator results.

II. Background/Review of related works

A typical mission involving multiple UUVs will have many distinct phases. Initially, the UUVs will be onboard their host vessel/vessels. Depending on the size of the UUVs involved in the mission and the goals of mission, there may be more that one host vessel deploying UUVs. After the UUV have been sea prepped (batteries charged, sensors calibrated, positioning systems initialized etc.) and mission prepped (destination and mission types defined) the UUVs will be put into the water. They will then form into a group and travel to the area of interest. Currently, we are assuming that there will be at least one vehicle that has an accurate positioning system on board. The others

will rely on it for corrections and updates. Upon getting to the area of interest, the UUVs will change into mission specific formations and execute their mission related goals. When the mission is complete they once again form a transit formation and journey back to their host vessel/vessels where upon arrival the collected data will processed and disseminated. Below is a brief summary of phases of a UUV mission:

1. Disembark.
2. Traveling formation creation.

3. Transit to Area of Interest.
4. Arrival
5. Survey formation creation.
6. Surveying of Area of Interest.
7. Traveling formation creation.
8. Transit back to home base.
9. Arrival and data downloading.

NRL's PNT (Positioning, Navigation and Timing) team has developed autonomous surveying techniques, phase 6 from the mission phases section above, for oceanographic vessels, both ships and submersibles. This effort has focused on the optimal deployment of swath sensor systems, i.e. sensors with a fixed angular swath width whose physical coverage varies with sensor altitude over the terrain and terrain shape. The primary development in this area is AutoSurvey [1] that generates next-line navigation waypoints based on the sensor data collected during the previous line. The objective of AutoSurvey is full sensor coverage in minimal time and simulation studies have shown time-savings of up to 30%.

Phases 2 and 3, traveling formation creation and transit to area of interest are very similar to their return counterparts, phases 7 and 8. Superficially the difference is that during phases 2 and 3 the vehicles are leaving the staging area, and in 7 and 8 they are returning, but the big difference is that during stages 2 and 3 the host vessels are near enough to detect and correct problems. In phases 7 and 8, the host vessels could be several miles from the UUVs, effectively leaving the UUVs on their own. In this paper we concentrate on the control system developed for phases 2 and 3, and while many of these ideas may work for phases 7 and 8, in this paper we do not attempt to apply or confirm them.

III. Discussion of formation maneuvering techniques explored in literature

Coordinating multiple autonomous vehicles moving in formation has recently become an active area of investigation in robotics, multi-agent systems, and control. A formation by a group of vehicles is a geometric configuration of the vehicles with specific relative position and orientations among them. For example, a group of 3 vehicles can form an equilateral triangle formation with the distance between any two vehicles equal 500 meters. Formation control deals with the problem of controlling the vehicles in a group so that the desired formation is obtained and maintained while the group is moving.

Roughly speaking, existing works on formation control fall into one of the following 4 categories:

1. Leader-follower approach: each vehicle (except a special leader vehicle) follows one or more designated leader vehicles with required relative position and orientation. The group in University of Pennsylvania has done extensive work in this leader-follower paradigm [3-6]. In [3], control laws have been developed for vehicles that follow one or two leaders. The stability of the control laws has been established.

2. The behavior-based approach [7] motor schemas (algorithms) are designed for each of the basic behaviors (such as keeping formation, avoidingobstacles, etc.), and the control command (heading and speed of the vehicle) is obtained by a weighted combination of the basic behaviors.

3. The potential field approach [8-10]: this approach associates an artificial potential field with a group of vehicles. The potential and the interaction force between a vehicle and its neighbor vehicle are dependent on the distance between these two vehicles. The artificial potentials should be designed in such a way that the desired group configurations is obtained at a global minima of the potential function. The movement of the vehicles of the group should be toward minimizing the potential.

4. Virtual structure approach [11]: consider the group of vehicles as forming a rigid structure, and move the group along a trajectory with control laws that minimize the deviation of each vehicle from the desired position in the virtual structure.

While some existing works emphasize the study of stability of the control laws (such as [3,6,8]), quite some others validate the control algorithms by simulations and some by actual robotic vehicle road testing [7]. The work in [12] considers the issue of reducing the communication need by observations/sensing in underwater vehicle formation control.

Compared with existing works about formation control, our work here has some distinctive features: First, we use a machine learning technique (genetic algorithms) to learn the control laws to move into (acquire) formation, and to keep (follow) formation. Existing works with machine learning in formation control are quite limited.

Some works studied use reinforcement learning to better estimate other agent's performance. In one of the most relevant works [13], a genetic algorithm is used to evolve neural network controllers for simulated "prey" creatures to learn a herding behavior avoiding predators. However, that work [13] does not address the issue of forming a particular geometric shape (line, tree, etc.). Second, we used a non-conventional neural network in the sense that the node functions themselves can be evolved to fit the problem at hand. Finally, the current work forms the first step toward a comprehensive, hierarchical system with learning capabilities for formation control for the PNT project.

IV. Approach

For this work, the overall goal was to endow the UUVs with the ability to create a formation and move to a destination in that formation. We recognized that seeking

and following leaders were two key abilities that would needed in order to realize our goals, so this paper discusses our pursuit of these sub-goals in the context of an adaptive system.

Fig. 1 below shows the conceptual diagram for a rudimentary adaptive system from which the implemented system was modeled. Based on internal states and sensory data from the environment, the Learn/Act box decides whether the system needs to improve its controllers. The Action Development box uses a machine learning technique to build/modify a controller. The Action Selection box uses controllers developed by the Action Development box. Its outputs go to the robots actuators, which in turn change the environment.

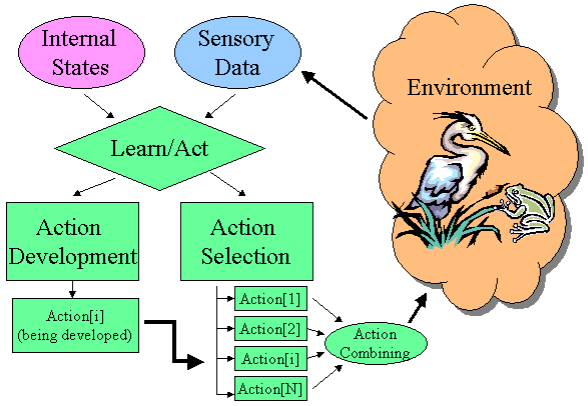

Fig. 1. This figure shows a conceptual diagram of an adaptive system whose main feature is the ability to improve its controllers.

Fig. 2 below shows the pared down system that serves as the initial controller system for formation creation and maintenance.

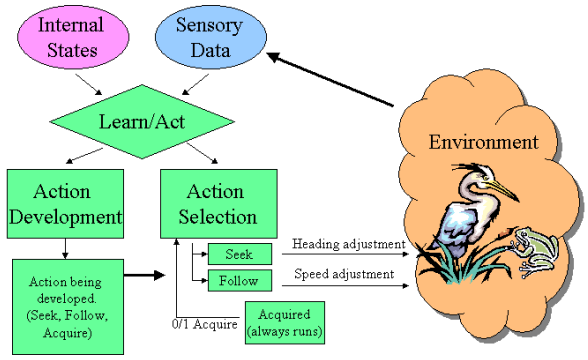

Fig. 2. This figure shows how the system in figure 1 was adapted for the formation maneuvering problem.

The thrust of the diagram in Fig. 2 is that by developing techniques to seek out a UUV and to follow a UUV various formations can be created and held. The effectiveness of the UUV can be increased if it can recognize which action is appropriate at the moment and have the ability to enhance the effectiveness of these actions for its current situation. Our system organization provides for this by using a neural controller to select what action to take, and what action to learn.

For the moment, we used a bottom up approach and decided to develop the key parts of the system in order to make sure that they would suffice for task at hand, formation creation and maintenance.

A multi-robot simulator, called robot school, was developed as the primary research tool. This simulator uses the Genetic Algorithm (GA) to grow feed forward neural network controllers for simulated UUV in a simulated environment. It uses LabView as a control and display engine and C as the calculation engine. The relevant input parameters of for the controllers, the lead UUV and the display of the simulated environment are managed by LabView, while the calculations of the various sensors, environmental effects, and control algorithms are done in a library written in C and linked to with LabView. Fig. 3 below shows the main control screen of the robot school program.

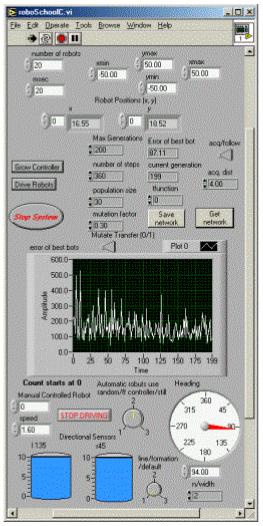

Fig. 3. This figure shows the main control screen for the robot school simulator. The graph in the middle shows the error level of controllers during training mode. The blue meters on the lower left of the screen show left and right acoustic sensor levels of the manually controlled robot. The large dial to the right is the directional control for the manually controlled robot.

Key features of the robot school simulator are:

1. UUV model. The robots are loosely modeled after the Underwater Navigation Control Lab (UNCL) lab robots (ActiveMedia Pioneer 2DX) in that they have the same speed characteristics, top speed of 1.8 m/sec is enforced.
2. Sensor model. See figure 4 below.
    a. Two omni directional sensors placed at 45 and 135 degrees on the robot at 2 meters from the center of the robot. The front of the robot is at 90 degrees.

b. Each robot has an ambient amount of noise and noise associated with the speed that it is moving. The faster it moves, the noisier.
c. Noise dissipates with the square of the distance between the source and the receiver.

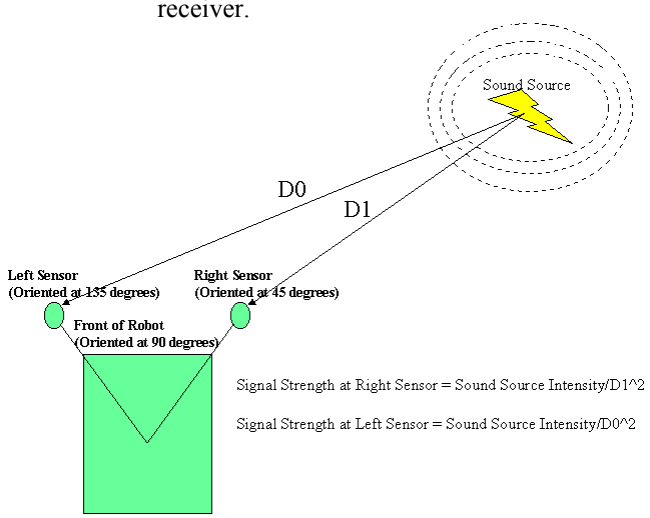

Fig. 4. This figure illustrates the robot acoustic sensor model as described in part 2 of the text.

3. Collision model. When robots collide they react according to a physical model with the following characteristics.
   a. Inelastic model.
   b. Robots modeled as round point masses.
   c. Collision radius is set at 1 meter.
4. Communication model.
   a. Each robot has a signature frequency that can be used to identify the robot.
5. GA implementation.
   a. Selection of parents. The parents of the next generation are picked from the top 20 percent of the current generation.
   b. Single crossover. Only one child results from the combining of chromosome sets.
   c. Fitness functions for Follow function and Seek function.
      i. The cost function for Follow minimizes distance between the leader and follower.
      ii. The cost function for Seek minimizes the distance between the leader and follower and maximizes the distance covered by the follower.
6. Feed forward network. See Fig. 5 below.
   a. The feed forward network has 2 nodes in the input layer, 4 nodes in the hidden layer, and 2 nodes in the output layer. It is fully connected. See figure 3 below.
   b. Trained by GA. Since we have no training data available, but do have a good idea of what the goals of each of the controllers should be, we use a GA to find the best set of weights for the feed forward networks.
   c. Transfer functions. We let the genetic algorithm select the type of transfer function that will be applied at each node of the hidden layer. It can select from linear (no transfer), discrete (if the input is greater than 0 then output is 1, otherwise it is –1), and sigmoid ($y = 1/(1+\exp(-v))$). So in this implementation, transfer functions in hidden layer can vary within a network.

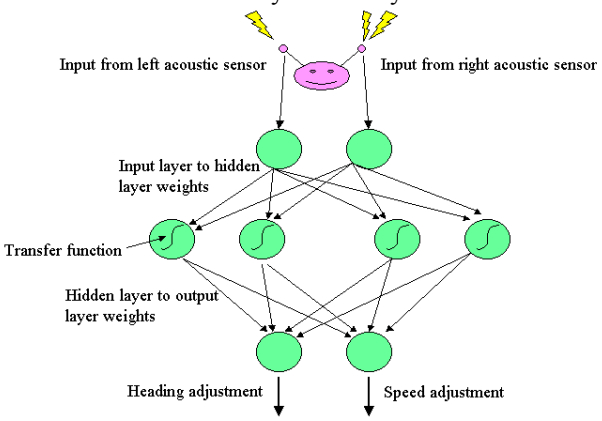

Fig. 5. This figure shows the geometry of the feed forward network used in both seek and follow modes. It has 2 nodes in the input layer, 4 in the hidden layer, and 2 in the output layer. Transfer functions in the hidden layer are selected by the GA.

In a typical session of the program, the user will use the genetic algorithm to grow either seek controller or a follow controller. Usually 200 generations will work well. After the controller has been grown, it can be tested by using a 2 UUV simulation. The user will control one of the UUVs and the just grown controller will control the other. If the controller appears to be good, it can be saved in a file for later use. See the example in Appendix A. In the simulated environment, all sensors and physical models apply to all UUVs. After the user is satisfied that his controller is working, i.e. if it is a follow controller the manually controlled UUV will be shadowed by the computer controlled UUV, a formation can be tested using the controller. Currently, both line and tree formations have been developed with lines working well with the current software. The tree formation needs the robots to be able to listen a specific set of robots, which the software is not currently set up to do. Currently a robot can be singled out and listened for, or all robots can be listened to.

V. Results and Discussion

For the follow function, the discrete transfer function works best if the network being generated is using all the same transfer functions in the hidden layer. This was somewhat of a surprise; the early favorite was the sigmoid function. Some networks with mixed transfer functions in the hidden layer have performed equally well, if not better. Note:

at this point in the development cycle, we do not a set of fixed exercises to test our grown networks on, so how good a particular network is somewhat subjective, hence the language "equally well if not better". It is however, very obvious when a network does not work well, so the subjective part occurs in degrees of "goodness". The above comments apply to the seek network as well.

In two robot situations, one leader and one follower, the behavioral difference between the two networks is that the follow network can keep the following robot in very close proximity to the leader, whereas the seek network tends to wander more and can loose track of the leader much easier. The seek network does have an advantage when it the leader robot has not been "locked" on to. When a follow network looses track of its leader, it becomes stationary, and when the leader returns it will actually repel the leader. The seek network is nearly opposite. The robot will keep moving, whether the leader is close or not, and when the leader is close enough to be detected it will move to it.

These behaviors are a direct result of the fitness functions. The fitness rule for the follow function rewards proximity to the leader. Whereas the fitness rule for the seek function rewards both proximity to the leader and area covered during the search for the leader.

VI. Formation Creation and Holding the Formation During Movement

With the current software, line and tree formations have been tested. Leader assignment for the line formation is as follows. Each robot follows the robot with id one less than its own id. Fig. 6 below shows the geometry of a robot line using this scheme.

Line Formation.

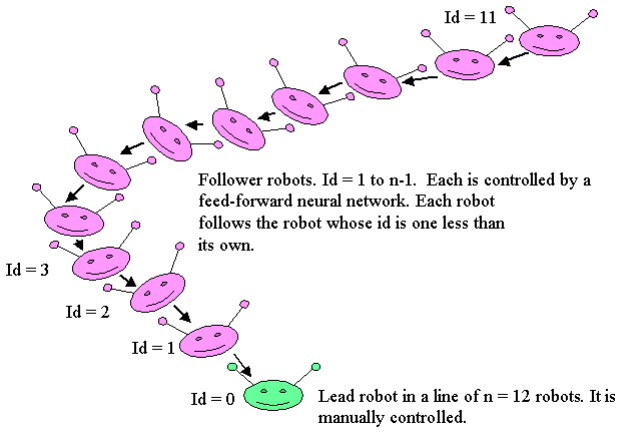

Fig. 6. This figure shows a robot line. The green robot is controlled manually from the robot school control screen. Using their sensors and feed-forward controllers, the other robots follow the robot whose id is one less than their own.

Tree formation leader assignment is similar. Each robot follows the robot with the id of its own divided by 2. Fig. 7 below illustrates the leader/follower relationship of the tree formation.

Modified tree Formation.

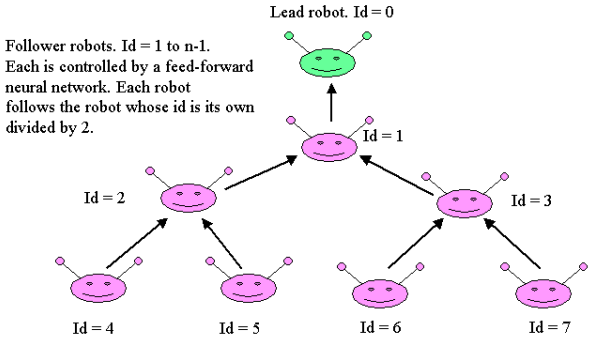

Fig. 7. This figure illustrates the leader/follower relationship in the tree formation.

The line formation is simplest because each robot follows its assigned leader. The formation remains stable as long as no robot looses its leader. On several occasions, lines of 20 robots have been maneuvered for up to 20 minutes in the simulator. Lines of up to 30 robots have been tested, but it is not easy to form up. Fig. 8 below shows robots forming a line of 20 robots, the green robot is the manually controlled leader.

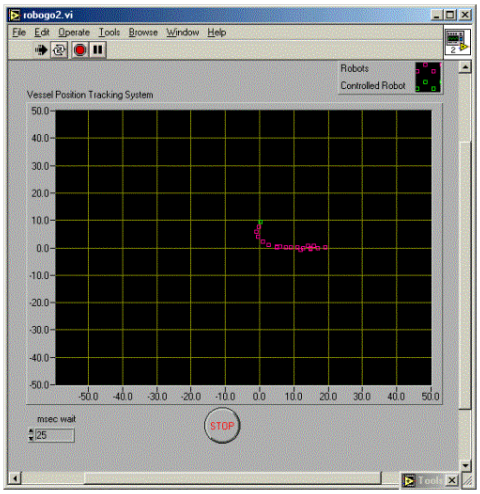

Fig. 8. This figure shows a line of 20 robots attempting to form a line. The green square is the leader robot.

The formation creation process occurs at the beginning of the simulation. At time zero each robot is placed in the simulation with a random heading and speed. The robots are placed along the X-axis in fairly close proximity to each other. At this moment the robots are milling around trying to get their bearings on their leaders. Several collisions occur because obstacle avoidance has not been worked into the fitness functions yet. In some cases the collisions may help in the line formation, because they help keep the robots bunched together long enough for each robot to find its leader. As the lead robot, usually robot 0, is manually driven away from the formation creation area, the robot line

emerges. Fig. 9, 10, and 11 show various lines of robots being maneuvered on the screen.

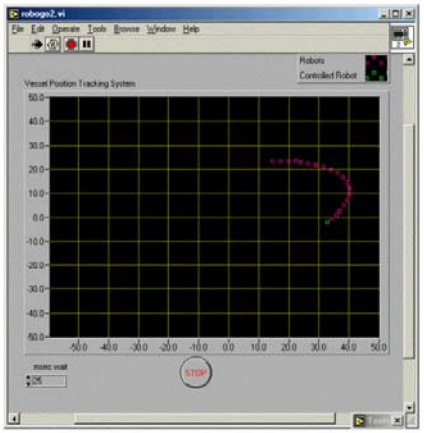

Fig. 9. Line of simulated robots/UUVs making a sweeping turn.

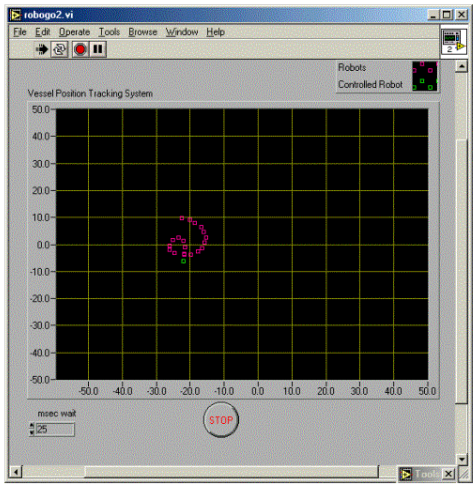

Fig. 10. This figure shows a loop in the line. As the line crosses itself the robots usually collide and bounce off each other. Usually they can remain "locked on to" their leaders keeping the line in tact.

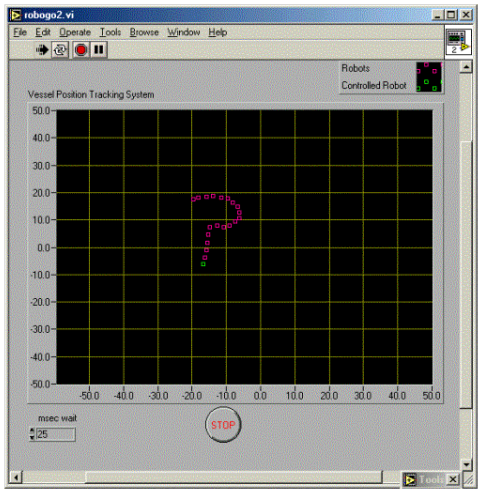

Fig. 11. This figure shows a formation that looks like a question mark. The sharp turn was the result of a collision that occurred when attempting to make a loop in the line.

Tree formations have been less successful. The current software supports listening to all robots and listening for one robot, but not listening for all robots, except one. Because of this, the tree leaf members of the tree formation tend to run into each other.

VII. Future Work

Immediate work will include improvement of the tree formation function. A parallel effort will be undertaken in the UNCL using the three resident mobile robots. On a more long-term schedule, the rest of figure 2 will be implemented and tested.

VIII. Conclusions

The simulator results show that biologically inspired methods such as feed-forward networks trained by a GA can be used to create, maintain and rebuild simple formations of simulated autonomous vehicles. This method could provide utility because it requires no precise geometric/navigation knowledge in the controller. It also has the benefit that it may work in situations where classic control methods are affected by discontinuities. The critical factor in the work so far has been creating appropriate cost functions, correct selection of transfer functions, and making sure that there exist a mapping from the inputs of the neural network to its outputs. While much work remains in making the system a self adapting system as in Figure 1, the work done so far shows that the low level functions that such a system would use can have utility when applied to the formation maneuvering.

Acknowledgments

This work was funded by the Office of Naval Research through the Naval Research Laboratory under Program Element 62782N. The mention of commercial products or the use of company names does not in any way imply endorsement by the U.S. Navy. Approved for public release; distribution is unlimited. NRL contribution number NRL/PP/7440-02-1025.

References

[1] B. Bourgeois and P. McDowell, "UUV Teams for Deep Water Operations", Underwater Intervention 2001, New Orleans, LA February 27 – March 2, 2002

[2] B. Bourgeois, A. Martinez, P. Alleman, J. Cheramie, J. Gravley, "Autonomous Bathymetry Survey System", IEEE Journal of Oceanic Engineering, vol. 24, no. 4, October 1999, pp 414-423.

[3] J. Desai, J.P. Ostrowski, V. Kumar (1998). "Controlling Formations of Multiple Mobile Robots", Proceedings of 1998 IEEE Int. Conf. on Robotics and Automation, 1998, pp. 2864-2869.

[4] J. Desai, V. Kumar, J.P. Ostrowski (1999). "Control of Changes in Formation For a Team of Mobile Robots",

Proceedings of 1999 IEEE Int. Conf. on Robotics and Automation, 1999, pp. 1556-1561.
[5] J. Desai, J.P. Ostrowski, V. Kumar (2001). "Modeling and Control of Formations of Nonholonomic Mobile Robots", IEEE Trans. on Robotics and Automation, vol. 17, no. 6, December 2001.
[6] R. Fierro, A. Das, V. Kumar, and J. P. Ostrowski (2001). "Hybrid Control of Formations of Robots," IEEE Int. Conf. on Robotics and Automation, Seoul, Korea, May 2001, pp. 157-162.
[7] T. Balch, R. Arkin (1998). "Behavior-Based Formation Control for Multirobot Teams", IEEE Trans. on Robotics and Automation, vol. 14, no. 6, December 1998.
[8] N.E. Leonard, E. Fiorelli (2001). "Virtual Leaders, Artificial Potentials and Coordinated Control of Groups", Proceedings of the 40th IEEE Conf. on Decision and Control, December 2001, pp. 2968-2973.
[9] T.R. Smith, H. Hanbmann, N.E. Leonard (2001). "Orientation Control of Multiple Underwater Vehicles With Symmetry-Breaking Potentials", Proceedings of the 40th IEEE Conf. on Decision and Control, December. 2001, pp. 4598-4603.
[10] H. Yamaguchi, T. Arai (1994). "Distributed and Autonomous Control Method for Generating Shape of Multiple Robot Group", Proceedings of IEEE Int. Conf. on Intelligent Robots and Systems, 1994, pp. 800-807.
[11] M.A. Lewis, K-H. Tan (1997). "High Precision Formation Control of Mobile Robots Using Virtual Structures", Autonomous Robots, 4, pp. 387-403.
[10] D. Stilwell, B.E. Bishop (2000). "A Framework For Decentralized Control of Autonomous Vehicles", Proceedings of the 2000 IEEE Int. Conf. on Robotics and Automation, April 2000, pp. 2358-2363.
[11] G. Werner, M. Dyer (1992). "Evolution of Herding Behavior in Artificial Animals", Proceedings of Int. Conf. on Simulation of Adaptive Behaviors, 1992.

APPENDIX A

The following text is the file that contains the weights and various parameters used to reconstruct a feed-forward network in the robot school.vi program.

Feed-forward network weights.
Network id
2
Transfer function (0-discrete, 1-linear, 2-discrete)
0
Number inputs
2
Hidden layer size
4
Number outputs
2
Weights from input layer to hidden layer node 0
-0.804000 0.343000
Weights from input layer to hidden layer node 1
-0.279000 0.497000
Weights from input layer to hidden layer node 2
0.965000 -0.719000
Weights from input layer to hidden layer node 3
0.511000 0.321000
Weights from hidden layer to output layer node 0
0.640000 -0.010000 -0.908000 0.852000
Weights from hidden layer to output layer node 1
-0.396000 -0.555000 0.440000 0.247000
Transfer functions.
0 0 0 0
End of network, good-bye.